状态压缩dp

2019年7月9日 20:47

一. 棋盘取数求最大和，要求任一被取的数的8邻域内无其他被取的数（CF101853E）

1. 先预处理每行的合法状态i能转移的下行合法状态j，则v[0]存储所有可行状态

```
for_(i,0,1<<16){
    if(i&(i<<1)) continue;
    for_(j,0,1<<16){
        if(j&(j<<1)) continue;
        if(j&(i<<1)) continue;
        if(j&(i>>1)) continue;
        if(j&i) continue;
        v[i].push_back(j);
    }
}
```

2. 再预处理每行所有可行状态的和

```
for_(i,0,n)
    for(auto t:v[0]){
        if(t>=1<<n) break;
        int sm=0;
        for_(j,0,n) if(t>>j&1) sm+=a[i][j];
        dr[i+1][t]=sm;      //注意此处一维空开下标0
    }
```

3. 最后再按行转移

```
for__(r,1,n) //预处理好dr，从空开的第0行，转到第n行，即为答案
    for(auto t:v[0]){   //上一行t
        if(t>=1<<n) break;
        for(auto tt:v[t]){  //这一行tt
            if(tt>=1<<n) break;
            dp[r][tt]=max(dp[r][tt],dp[r-1][t]+dr[r][tt]);
            ans=max(ans,dp[r][tt]);
        }
    }
```

二. Mondriaan's Dream（POJ2411）

1. 题：把n行m列的棋盘分成若干1*2的长方形，求方案数
2. 引：令1表示该方块和下一行的相连，则0表示与上一行的1相连或与左或右的0相连；按行分割棋盘，每列的10有在二进制下对应的权值，则可将每个i行分割成若干个状态j；令f[i][j]=i行状态对应j时，前i行的可行方案总数
3. 引：状态j按位与状态k得0时，表示上下相连完全合法；j按位或k得到的连续零的长度为偶数个时表示左右相连也完全合法，这个比较难判断，可以事先预处理哪些状态是合法的
4. 结：f[i][j]=[j&k是0] [j|k合法] Σ f[i-1][k]；初值只有f[0][0]=1；答案f[n][0]

```
memset(f, 0, sizeof(f));
memset(lgl, 0, sizeof(lgl));
f[0][0] = 1;
for_(i,0,1<<m){ //预处理每个状态i是否符合不存在奇数个0的条件
    bool cnt_0 = 0;
    bool odd_0 = 0;
    for_(j,0,m)
        if(i>>j &1)       //需要重新数零，顺便判断一下之前数到的0是不是奇数
```

```
            个
                odd_0 |= cnt_0,
                cnt_0 = 0;
            else    //继续数现在有几个0
                cnt_0 ^= 1;
        odd_0 |= cnt_0;          //可能最后一串0没有被计入
        lgl[i] = !odd_0;
}
for__(i,1,n)
        for_(j,0,1<<m)
            for_(k,0,1<<m)
                if(!(j&k) && lgl[ j|k ])
                    f[i][j] += f[i-1][k];
```